Figure 1

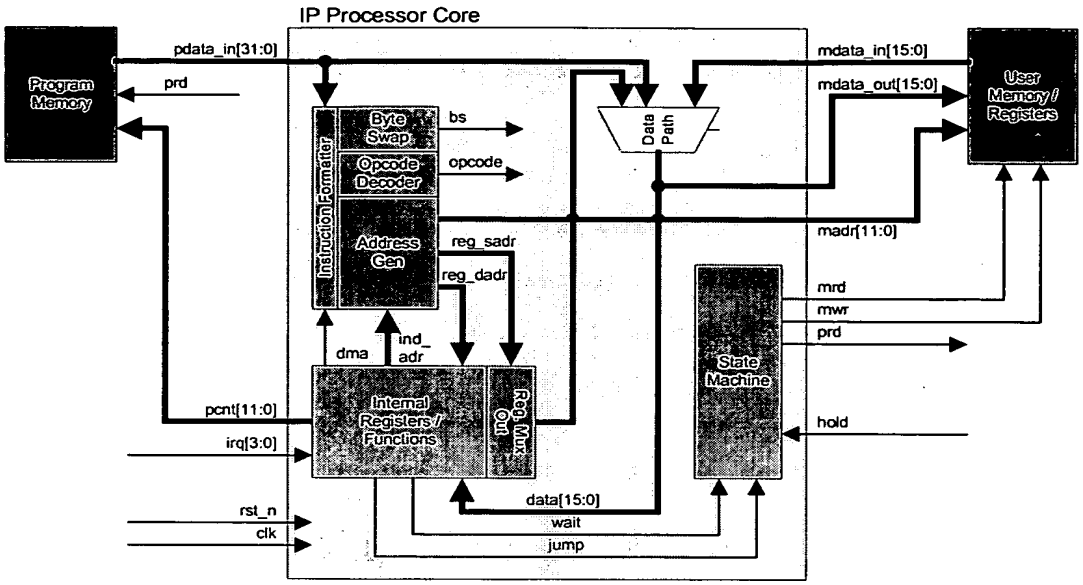

IP Processor Core
Program Memory
pdata_in[31:0]
prd
pcnt[11:0]
irq[3:0]
rst_n
clk
Instruction Formatter
Byte Swap
Opcode Decoder
Address Gen
bs
opcode
reg_sadr
reg_dadr
dma
ind_adr
Internal Registers / Functions
Reg Mux
Data Path
mdata_in[15:0]
mdata_out[15:0]
madr[11:0]
mrd
mwr
prd
State Machine
hold
data[15:0]
wait
jump
User Memory / Registers

Figure 2

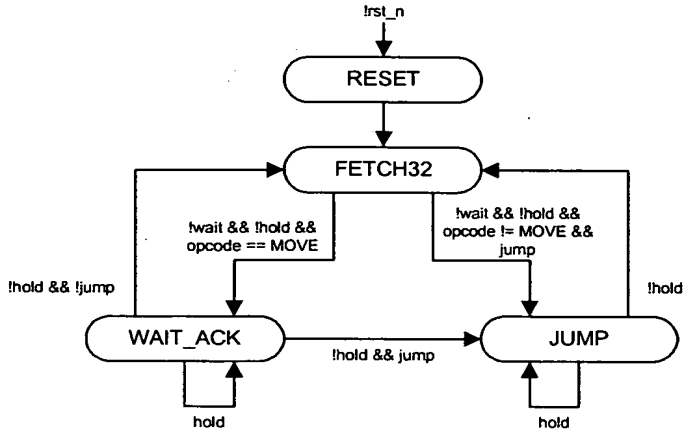


Figure 3

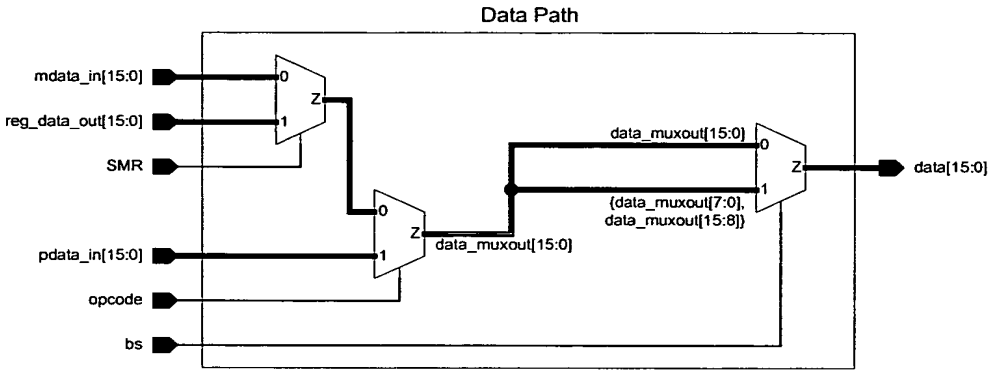


Figure 4

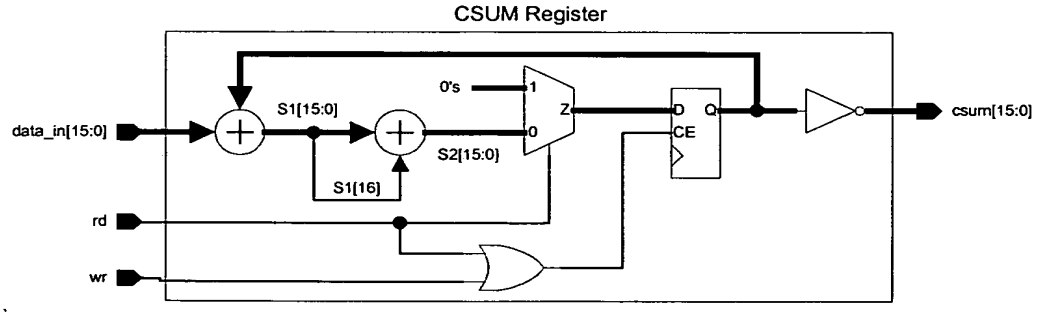


Figure 5

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	DMR	Destination Address (DA)												DBS	x	x	SMR	Source Address (SA)													SBS

DMR	Selects the destination region ('0' for User Memory; '1' for Registers).
DA	Destination Address, in word.
DBS	The data word is byte swapped when DBS ^ SBS = 1.
SMR	Selects the source region ('0' for User Memory; '1' for Registers).
SBS	The data word is byte swapped when DBS ^ SBS = 1.

Figure 6

1	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	DMR	Destination Address (DA)												DBS	Constant Word (CW)																

DMR	Selects the destination region ('0' for User Memory; '1' for Registers).
DA	Destination Address, in word.
DBS	The data word is byte swapped when SBS = 1.
CW	Constant word to be loaded into the destination address.

Figure 7

REG_A	**General Purpose Register A**	
Access	Read/Write. Auto-Updated.	
Bit	**Name**	**Description**
B[15 :0]	A[15:0]	General Purpose Register A.

Figure 8

REG_B	**General Purpose Register B**	
Access	Read/Write. Auto-Updated.	
Bit	**Name**	**Description**
B[15 :0]	B[15:0]	General Purpose Register B.

Figure 9

PCNT	Program Counter	
Access	Write Only. Auto-Updated.	
Bit	**Name**	**Description**
B[15]	IE	If Equal jumping condition.
B[14]	IG	If Greater jumping condition.
B[13]	IN	If Not jumping condition.
B[12]	Reserved	Unused – always write "0".
B[11:0]	PCNT[11:0]	Program Counter. On a write, Program Counter is updated with PCNT[11:0] value if the following condition is met: case ({IE, IG}) 2'b00: 1 2'b01: IN ^ ((MASK & A) > (MASK & B)) 2'b10: IN ^ ((MASK & A) == (MASK & B)) 2'b11: IN ^ (((MASK & A) == (MASK & B)) || ((MASK & A) > (MASK & B))). After every program memory fetching, Program Counter is incremented by 1.

Figure 10

RETA	Return Register	
Access	Read Only. Auto-Updated.	
Bit	**Name**	**Description**
B[15:12]	Reserved	Unused.
B[11:0]	RETA[11:0]	Return Register. On a write in the Program Counter with IE = 0, IG = 0 and IN = 1, the Return Register is loaded with PCNT[11:0] value.

Figure 11

MASK	Mask Register	
Access	Read/Write.	
Bit	**Name**	**Description**
B[15 :0]	MASK[15:0]	Mask register used for some operations like comparison, bit setting and bit clearing. By default this register is set to FFFFh.

Figure 12

WAIT	Wait Register	
Access	Read/Write. Auto-Updated.	
Bit	**Name**	**Description**
B[15]	T1	Set this bit to one makes the processor waiting for timer 1 reaches zero. If more than one bit is set in the WAIT register, the processor stops waiting on the first event. Read value is one when timer 1 is equal to zero.
B[14]	T0	Set this bit to one makes the processor waiting for timer 0 reaches zero. If more than one bit is set in the WAIT register, the processor stops waiting on the first event. Read value is one when timer 0 is equal to zero.
B[13:4]	Reserved	Unused – always write "0".
B[3:0]	I[3:0]	Set one bit to one makes the processor waiting for the corresponding interrupt. If more than one bit is set in the WAIT register, the processor stops waiting on the first event. Read this register shows which interrupt occurred and clears I[3:0] bits.

Figure 13

TIMER0	Timer 0 Register	
Access	Write Only. Auto-Updated.	
Bit	**Name**	**Description**
B[15 :0]	TIMER0[15:0]	Timer 0 Register. Write a non-zero value sets and starts the timer. It is decremented on every clock cycle. When it reaches zero, the timer is stopped and T0 bit is set in WAIT register.

Figure 14

TIMER1	Timer 1 Register	
Access	Write Only. Auto-Updated.	
Bit	**Name**	**Description**
B[15 :0]	TIMER1[15:0]	Timer 1 Register. Write a non-zero value sets and starts the timer. It is decremented on every clock cycle. When it reaches zero, the timer is stopped and T1 bit is set in WAIT register.

Figure 15

CSUM	CheckSum Adder Register	
Access	Read/Write.	
Bit	**Name**	**Description**
B[15 :0]	CSUM[15:0]	CheckSum Adder Register. On a write, the 16-bit one's complement sum is computed from the previous value and the written value. On a read, the read value is inverted (~CSUM[15:0]) and it is reset to zero.

Figure 16

DMA	DMA Register	
Access	Write Only. Auto-Updated.	
Bit	**Name**	**Description**
B[15:12]	Reserved	Unused – always write "0".
B[11:0]	LEN[11:0]	DMA Length Register. Write any value in the LEN field starts a DMA of LEN bytes (4096 if value = 0) from address contained in A[13:0] to address contained in B[13:0]. Bit 13 of each address register indicates if it's from/to register (0) or memory (1). During the DMA, memory addresses are incremented and register addresses are not incremented.

Figure 17

PCNT Flags			Comparators		Result	
IE	**IG**	**IN**	**eq**	**gt**	**jump**	**call**
0	0	0	X	X	1	0
0	0	1	X	X	1	1
0	1	IN	X	gt	IN ^ gt	0
1	0	IN	eq	x	IN ^ eq	0
1	1	IN	eq	gt	IN ^ (eq | gt)	0

Figure 18

Macro	Opcode	Arg0	Arg1	Description
JMP Addr[11:0]	LOAD	K_PCNT_ADR	{4'h0, Addr[11:0]}	Inconditional Jump at Addr[11:0].
JEQ Addr[11:0]	LOAD	K_PCNT_ADR	{4'h8, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) == (B & MASK).
JGT Addr[11:0]	LOAD	K_PCNT_ADR	{4'h4, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) > (B & MASK).
JGE Addr[11:0]	LOAD	K_PCNT_ADR	{4'hC, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) >= (B & MASK).
JNE Addr[11:0]	LOAD	K_PCNT_ADR	{4'hA, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) != (B & MASK).
JLE Addr[11:0]	LOAD	K_PCNT_ADR	{4'h6, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) <= (B & MASK).
JLT Addr[11:0]	LOAD	K_PCNT_ADR	{4'hE, Addr[11:0]}	Jump at Addr[11:0] if (A & MASK) < (B & MASK).
CALL Addr[11:0]	LOAD	K_PCNT_ADR	{4'h2, Addr[11:0]}	Inconditional Jump at Addr[11:0] and store PCNT register value in RET register. Used to call a sub-routine and return from it with the RET instruction.
RET	MOVE	K_PCNT_ADR	K_RETA_ADR	Inconditional Jump at address stored previously in RETA register. Used to return from a sub-routine CALL instruction.

Figure 19

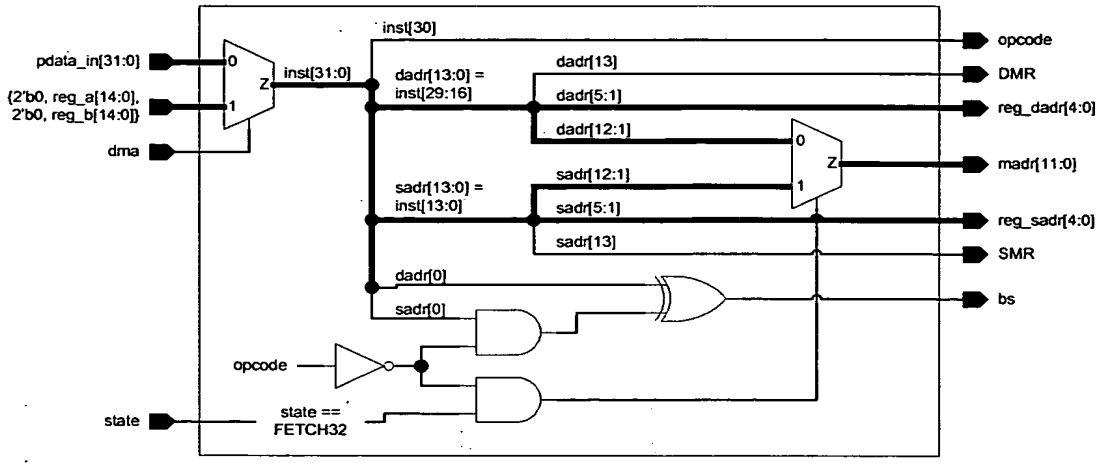


Figure 20

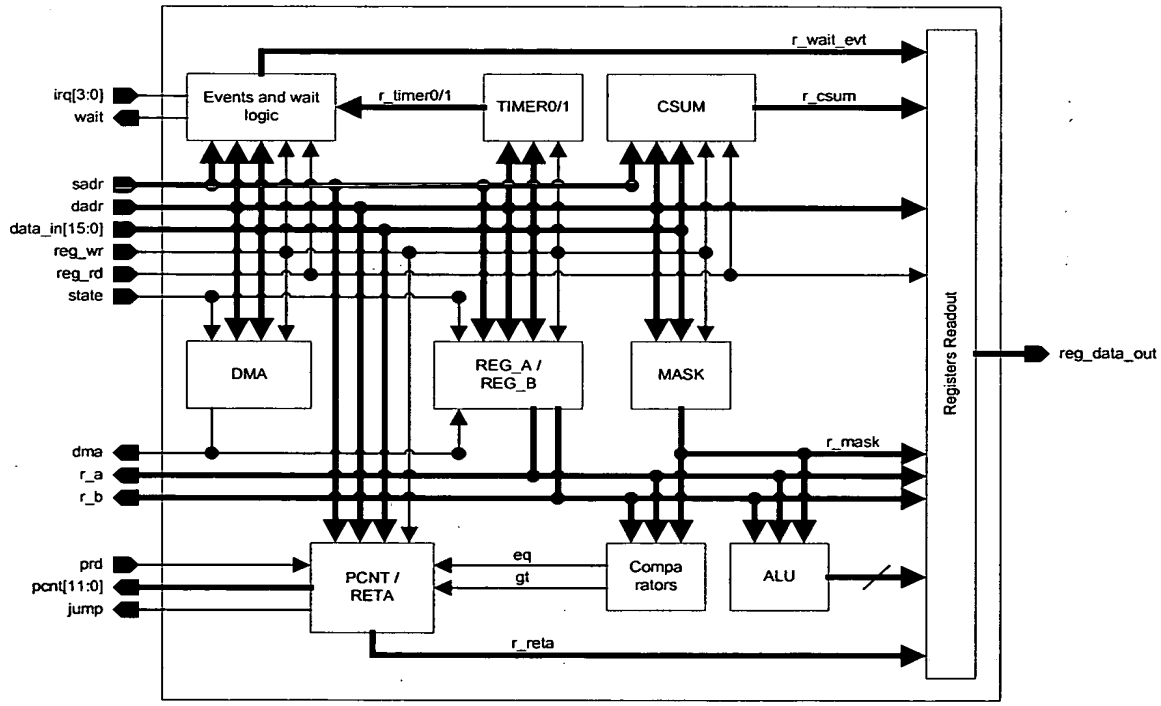
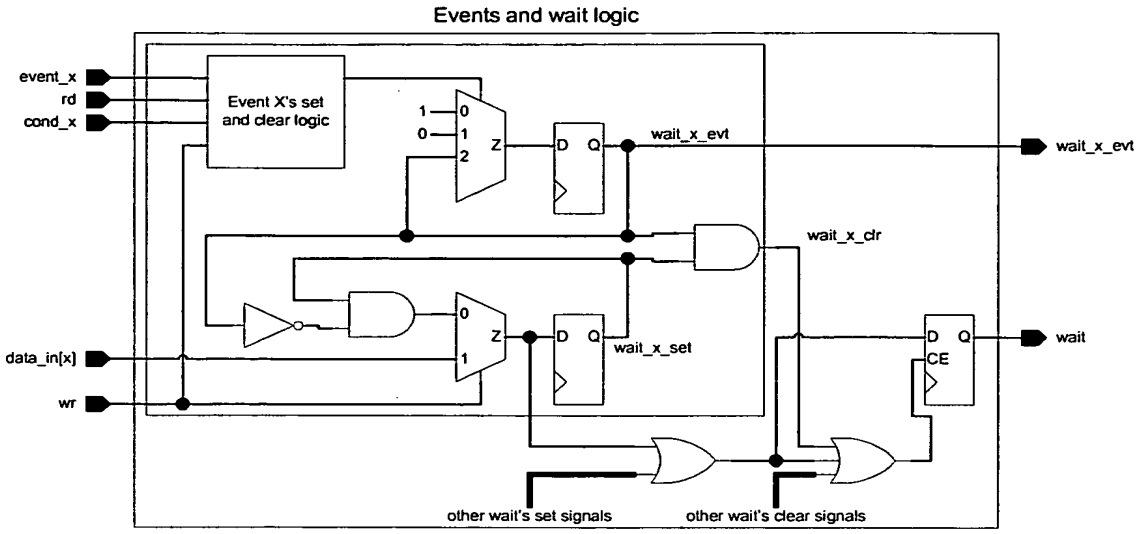


Figure 21


Events and wait logic
event_x
rd
cond_x
Event X's set and clear logic
1
0
0
1
2
Z
D
Q
wait_x_evt
wait_x_evt
wait_x_clr
0
1
Z
D
Q
wait_x_set
data_in[x]
wr
D
Q
CE
wait
other wait's set signals
other wait's clear signals